PRELIMINARY USER'S MANUAL

TRS-80
MICRO COMPUTER SYSTEM

Catalog Numbers:

26-1001
26-1201
14-841

©*Copyright 1977, by Radio Shack, A Division of Tandy Corporation, Fort Worth, Texas 76107, U. S. A.*

TRS-80 PRELIMINARY MANUAL

This User's Manual is not intended to be the final, complete one. Since we know you are eager to start using your Microcomputer system, we decided not to delay your shipment any further ... and to send you this Preliminary Manual (along with some game tapes and other materials).

SETTING UP THE SYSTEM

Carefully unpack the system. Remove all packing material. Be sure you locate all cables, papers, tapes, etc. Save the packing material in case you need to transport the system.

Connecting the Video Monitor and Keyboard:

1. Connect the power cord from the Video Monitor to a source of 120 volts, 60 Hz AC power. Note that one prong of the AC plug is wider than the other -- the wide prong should go into the widest slot of the AC socket.

   NOTE: If you use an AC extension cord, you may not be able to plug the Monitor's power cord in. Do not attempt to force this wide prong into the extension cord; use a wall outlet if at all possible.
2. Connect the power cord of the Power Supply to a source of 120 volts, 60 Hz AC power.

3. Connect the gray cable from the front of the Video Monitor to the VIDEO jack on the back of the Keyboard Assembly. Take care to line up the pins correctly (the plug fits only one way).
4. Connect the gray cable from the Power Supply to the POWER jack on the back of the Keyboard Assembly. Again, take care to mate the connection correctly.

Connecting the Cassette Recorder:

You do not need to connect the Cassette Recorder unless you plan to record programs or to load taped programs into the TRS-80.

1. Load batteries into the CTR-41 as described in the Manual. Or make connections for 120 volt AC power.
2. Connect the short cable (DIN plug on one end and 3 plugs on the other) to the TAPE jack on the back of the Keyboard Assembly. Be sure you get the plug to mate correctly.
3. The 3 plugs on the other end of this cable are for connecting to the CTR-41.
   A. Connect the black plug into the EAR jack on the side of the CTR-41. This connection provides the output signal from the CTR-41 to the TRS-80 (for loading Tape programs into the TRS-80).
   B. Connect the larger gray plug into the AUX jack on the CTR-41. This connection provides the recording signal to record programs from the TRS-80 onto the CTR-41's tape.
   Also, plug the Dummy Plug (provided with the CTR-41) into the MIC jack (this disconnects the built-in Mic so it won't pick-up sounds while you are recording tapes).

C. Connect the smaller gray plug into the REM jack on the CTR-41. This allows the TRS-80 to remotely control the CTR-41's motor (turn tape motion on and off for recording and playing tapes).

Notes On Using The Recorder

There are a number of things you should be aware of as you use the Cassette Tape System:

1. To Play a tape (load a taped program into the TRS-80), you must have the CTR-41's Volume control set to 7 to 8 . Then press the CTR-41's PLAY key and then type CLOAD on the TRS-80 and ENTER this command. This will start the tape motion. An * will appear on the top line of the Monitor; a second * will blink, indicating the program is loading. When loading is done, the TRS-80 will automatically turn the CTR-41 off and flash READY on the screen. You are then ready to RUN the program (type in RUN and hit ENTER).
2. To Record a program from the TRS-80, press the CTR-41's RECORD and PLAY keys simultaneously. Then type CSAVE on the TRS-80 and ENTER this command. When the program has been recorded, the TRS-80 will automatically turn the CTR-41 off and flash READY on the screen. Now you have your program on tape (it still is in the TRS-80 also). Many computer users make a second or even a third recording of the tape, just to be sure they have a good recording.

NOTE: To load the full 4K of RAM in the TRS-80 takes less than 3 minutes of tape. Short programs will take only a few seconds of tape.

3. Use the CTR-41's Tape Counter to aid you in locating programs on tapes.
4. For best results, use Radio Shack's special 10 minute Computer Tape Cassettes (especially designed for recording computer programs). If you use standard audio tape cassettes, be sure to use top quality, such as Realistic SUPERTAPE or CrO_2. Keep in mind that audio cassettes have lead-ins on both ends (blue non-magnetic mylar material) -- you can not record on the leader portion of the tape. Advance the tape past the leader before recording a program.
5. When you are not going to use a CTR-41 for loading or recording programs, do not leave RECORD or PLAY keys down (press STOP).
6. To REWIND or FAST-Forward a cassette, you must disconnect the plug from the REM jack (with REM jack connected, the TRS-80 controls tape motion).
7. If you want to save a taped program permanently, break off the erase protect tab on the cassette (see CTR-41 Manual).
8. Do not expose recorded tapes to magnetic fields. Avoid placing your tapes near the Power Supply.
9. To check if a tape has a program recorded on it, you can disconnect the plug from the EAR jack (also disconnect the REM plug so you can control the CTR-41 with the keys) and Play the tape; you'll hear the program material from the speaker.

TURNING THE SYSTEM ON

Turn on the Video Monitor by pressing the POWER button. Turn on the TRS-80 Keyboard by pressing the POWER button on the back (next to the POWER jack); the red LED just to the right of the Keyboard should light up and the screen should show READY. Adjust C (contrast) and B (brightness) controls on the front of the Monitor for the sharpest display. Set Brightness so the background is gray and the words are white. Do not set Brightness too high.

If Monitor does not show READY, press the Keyboard's POWER switch off, wait 10 seconds, and then turn it on again. (Pressing the power switch slowly can cause the machine to "get confused". Turn it on with a quick push.)

Now type in P.M. and hit ENTER. This is a test to see that the Computer powered-up properly. The display should read:

```
P.M.
3583
```

If the number is not 3583, turn the Keyboard's POWER switch off, wait 10 seconds, and then turn it on again. Repeat the test and verify that the number is 3583. (It's good practice to perform this simple test whenever you turn on the TRS-80. Always type NEW and ENTER before performing the test.)

> NOTE: There is a Reset button inside a door at the left rear of the Keyboard assembly. This Reset button can be used to unlock a looping program or if the TRS-80 does not turn off a cassette or in other such abnormal program situations.

GETTING STARTED

This Preliminary User's Manual is not intended to be a detailed, complete course in Radio Shack's Level I BASIC. For that you'll have to wait for the complete Manual. However, we will try to give you enough help here so you can exercise your Computer a little before getting into the meat.

IMPORTANT NOTE: Each line of a program must be entered into the Computer. You do this by pressing ENTER key. To correct a typing error, press the backspace key (←); repeat as necessary to erase the error. If you have already ENTERed a line and notice an error, type the same line over again -- the Computer always takes the last line typed in.

Let's try the "calculator mode" first. It's the easiest. Type the following:

```
   PRINT 5-2 (hit ENTER)
```

The Display will show

```
   3

  READY
 >_
```

NOTE: With the TRS-80, there is no need to use SHIFT to type capital letters (all letters are capital). SHIFT is needed only for the keys which have two symbols.

This means the Computer is ready for more. (From now on, we won't repeat the READY and >_ .)

Now try

```
   PRINT 2/3, 2*8 (Hit ENTER. Note you must use PRINT to receive a
                   display.)
```

```
.666667         16
```

(the answer to the first -- 2 divided by 3 -- is given first. The second answer -- 2 multiplied by 8 is given second, moved over 16 spaces because of the comma.)

You can write a program for these same functions. A program must have a number for each line. Try this:

```
10 PRINT 14.5 + 3
20 PRINT 6.1 - 4.2
30 PRINT 7/22
40 PRINT 8*2.1
```

(Remember to hit ENTER at the end of each line.)

This is a program. Each line begins with a number. To execute (run) the program, type RUN and hit ENTER. The screen will display --

```
17.5
1.9
.318182
16.8
```

Press CLEAR and the screen will be blank except for the little cursor (_) at the upper left hand corner.

If you want to recall the program that is in the Computer, type LIST and the program will be displayed on the screen.

Use LIST any time you want to check the program that is stored in the Computer. You can use LIST to check a program that you have loaded from a cassette tape.

If you want to delete a line, all you have to do is type that line number and ENTER it. That replaces the existing program line number with a blank line. Try it on the program you have in the Computer. Then LIST the program and see what has happened.

Notes on Line Numbers:

You don't have to type line numbers in sequence. The Computer automatically puts lines in sequence.

You can add line numbers at any time (as long as you have a >_ on the screen).

Don't number lines in direct sequence; leave blank numbers so you can add more to your program later on. Example, use 10, 20, 30, etc.; then later on as you work with the program, you can add 15, 25 ... (or any or all numbers between the lines you have used).

More on PRINTing:

Often you will want to put some wording in with your program, to explain the meaning of calculations. Try this:

```
PRINT "THE APPROXIMATE VALUE OF PI IS", 22/7 (Hit ENTER)
```

The screen displays

```
THE APPROXIMATE VALUE OF PI IS  3.14286
```

The comma (,) moved the number to the next 16 character frame of the printing field. Try a semicolon instead and see what happens.

Try the following:

```
PRINT 1,2,3,4              (Remember to hit ENTER)
1               2               3               4
```

Now try:

```
PRINT 1;2;3;4              (Don't forget to ENTER it)
1  2  3  4
```

The commas space out the printing into columns 16 characters wide. The semicolons give just one extra space. Try some more till you get the feel of how and where the printing will be.

Now, type RUN. What happened? The screen came back with

```
17.5

1.9

.318182

16.8
```

Why? The program that we typed in earlier is still in the Computer? Not sure? Type LIST and see.

How to clear out previous programs? Of course we could erase each line (by typing in line numbers one at a time and ENTERing them). But there is an easier way. Type NEW. This clears the memory of all previous programs and you're ready to enter more. Always type NEW before loading new programs from cassettes or typing in new programs.

Type

```
10 PRINT "HELLO THERE. I AM YOUR NEW TRS-80 MICROCOMPUTER!"
```

RUN this one-line program:

```
HELLO THERE. I AM YOUR NEW TRS-80 MICROCOMPUTER!
```

Add another line:

```
20 PRINT "YOU CALLED, MASTER. DO YOU HAVE A COMMAND?"
```

Now RUN the program. You get both lines.

```
HELLO THERE. I AM YOUR NEW TRS-80 MICROCOMPUTER!

YOU CALLED, MASTER. DO YOU HAVE A COMMAND?
```

Now try it again. This time try talking to it; type in YES.

```
WHAT?
```

The Computer is telling you it does not know what you want. You see, you must talk to it in its own language. The TRS-80 understands the BASIC language (a simple language for Computers).

Add another line to your program:

```
   15 PRINT
```

What is that going to add? Just an empty line. RUN it and you'll see that the lines are separated; looks better now.

Let's add a REMark statement at the beginning. A REMark statement is not part of the program, it is just like a note or reminder to you.

```
    5 REM ** THIS IS MY FIRST COMPUTER PROGRAM **
```

RUN the program and you'll see that line 5 had no effect. Now pull a list (type LIST) and sure enough, line 5 is there.

Big & Little Numbers

The TRS-80 uses scientific notation to handle very large and very small numbers. For example, the number 5,000,000 becomes 5E+06; this means the number 5 followed by 6 zeros. The number .000,005 becomes 5E-06; this means 6 decimal places to the left of 5. Scientific notation is a very common method for listing very large and very small numbers (they are much easier to work with this way).

Sequence of Math Operators

Your TRS-80 uses the 4 fundamental math functions. You also can use parentheses in your math formulas. The following rules apply for ( ) and sequence of Math Operators:

1. All operations inside parentheses are performed first.
2. All operations buried deepest within ( ) are performed first.

   IE. If you have operations nested inside ( ), which are nested inside more ( ), the innermost ones are solved first.

3. Then, the Computer figures multiplication and division operations ... in sequence from left to right. Then it starts at the left again and performs addition and subtraction. IE. If you have a "chain" calculation which has multiplication, division, addition and subtraction, the Computer does multiplication and division first, then addition and subtraction.
4. A problem listed as (X)(Y) will not tell the Computer to multiply. X * Y is for multiplication.

If you want to be sure your problems are calculated correctly, use ( ) around operations which you want performed first.

NOTE: INT, RND and ABS functions are performed before multiplication and division.

Conditional Branching

Type in the following program.

```
10 A = 5
20 IF A = 5 THEN 50
30 PRINT "A DOES NOT EQUAL 5"
40 END
50 PRINT "A EQUALS 5"
```

RUN this program. The screen will show:

```
A EQUALS 5
```

In line 10 you assigned the value of 5 to A. Line 20 checked the value of A, found it was 5 and branched to 50 which resulted in the printing. Now, change line 10 to give A some other value and RUN it. The display will show:

```
A DOES NOT EQUAL 5
```

Of course, you may not want to change line 10 each time you want to give the Computer a value of A. Let's introduce another BASIC statement: INPUT. Add the following lines:

```
 5 PRINT "THE VALUE I WISH TO GIVE A IS"
10 INPUT A
```

RUN the program. The screen will show:

```
THE VALUE I WISH TO GIVE A IS
?_
```

The Computer is asking you to give it a value for A. Type a number and ENTER it. See what happens now. Try other numbers.

Now change line 5 as follows:

```
 5 INPUT "THE VALUE I WISH TO GIVE A IS";A
```

Delete line 10 (type a line 10 and ENTER it with nothing else).

Run the program now and see what happens:

```
THE VALUE I WISH TO GIVE A IS?_
```

Type a number; ENTER it and see if the program works the same.

You made a number of changes with lines 5 and 10:

A. Changed PRINT to INPUT.

B. Changed A to be on the same line as the printing info.

C. Eliminated one line.

This illustrates one of the ways you can shorten a program. (We'll have lots more on this in the final version of the User's Manual.)

The conditional branching we tried above (IF .... THEN) can be used for other purposes too. We can use a statement GOTO, telling the Computer to go to a specific line number and continue from there. Or, we can tell the Computer to go to a sub-routine program (ON ... GOSUB 32000). Branching becomes a very powerful tool in the hands of an experienced programmer (hang in there, you'll become one).

Counting and Looping

Try this.

```
   10 A = 0
   20 A = A + 1
   30 PRINT A
   40 GOTO 20
```

RUN it and see what happens.

To stop the Computer, hit BREAK. To restart it, type CONT (for CONTinue). Hit CLEAR and see what happens.

Now change line 30 to end with a comma. RUN it. Then try a semi-colon. RUN it.

Add a line for branching.

```
   35 IF A = 100 GOTO 100
  100 END
```

RUN it. Or, you could combine lines 35 and 100 as follows:

```
   35 IF A = 100 END
```

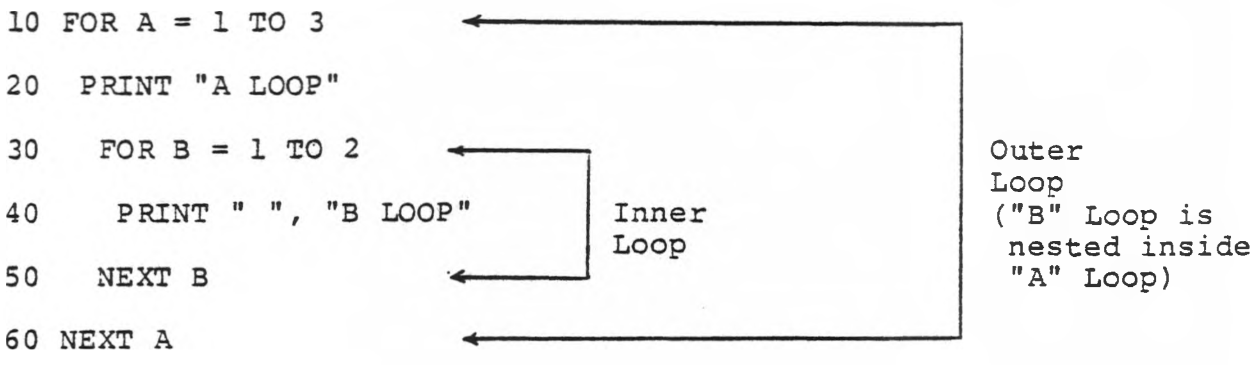
Try this:

```
10 FOR N = 1 TO 10
20 PRINT N*N
30 NEXT N
```

RUN this simple program. What happened? We set up a FOR-NEXT loop. Line 10 set a series of values for N (from 1 to 10). Line 20 tells the Computer to multiply N times N (the square of N). Line 30 tells the Computer to take the next N. This program goes through 10 loops. Change line 10 for more loops. Change line 20 to end with a comma or semi-colon.

NOTE: A FOR-NEXT loop automatically increases the number by 1, unless you tell it differently. Change 10 to 10 FOR N = 1 TO 10 STEP 2. RUN it.

Loops can be combined -- as long as the loops nest within each other (ie. finish one loop before going back to the first). Example: (type in NEW to clear out the memory):



```
10 FOR A = 1 TO 3
20  PRINT "A LOOP"
30   FOR B = 1 TO 2
40    PRINT " ", "B LOOP"
50   NEXT B
60 NEXT A
```

Note how we've indented info for the B Loop so you can see it easier. This technique is a great aid to proper programming.

RUN the program.

The inner loop must be completed before going back to the outer loop. Loops are extremely useful tools for computer programs. We'll cover them in great detail in the final User's Manual.

READ and DATA

Up to now we've looked at two ways to enter numbers into a program. One is by assigning a value to a letter (A = 5). Another is with the INPUT statement. Another method is with DATA.

Try this:

```
10 DATA 1,2,3,4,5
20 READ A,B,C,D,E
30 PRINT A;B;C;D;E
```

RUN this program.

NOTE: DATA statements can have values assigned with commas separating them; same with READ statements.

DATA lines are always read from left to right by READ statements. It does not matter where DATA statements come in the sequence of a program. Move line 10 to line 25. Move it to line 10000. Works the same in any case.

TAB Function

The TAB function is just like the Tab on a typewriter. It tells the Computer to start printing at a specified number of spaces in from the left. Try this:

```
10 PRINT TAB(12),"@@@@"
20 PRINT TAB(32),"####"
```

RUN it.

```
            @@@@

                                ####
```

PRINT AT Function

The screen can be divided up into 1024 individual squares. These squares are used for the Graphics functions, but you can also use them with a PRINT function.

```
PRINT AT 650, "TRS-80 MICROCOMPUTER"
```

ENTER this and see where the message is printed. There is a pad of paper with markings showing the location of these squares on the screen. Examine these pages and soon you'll be able to set up print statements for almost any part of the screen.

Graphics Functions

There are three Graphics statements. SET, RESET and POINT. SET turns on a square on the screen. RESET turns a square off. POINT checks for the presence of a signal at a square on the screen. Locations on the screen are defined with coordinates X and Y. X can be any value from 0 to 127. Y can be any value from 0 to 47.

Try this:

```
10 X = 27
20 Y = 27
30 SET (X, Y)
```

RUN it. Change the values of X and Y. (NOTE: Coordinates are not limited to X and Y. Try any letter combinations.)

Draw some lines with the SET function.

```
10 X = 32
20 FOR Y = 0 to 47
30 SET (X, Y)
40 NEXT Y
```

RUN it.

Change the value of X. Also, try changing the loop for Y (0 TO 10; 10 TO 20; 0 TO 47 STEP 5).

Try this program:

```
 10 FOR Y = 0 TO 10
 20  FOR X = 0 TO 40
 30   SET (X, Y)
 40  NEXT X
 50 NEXT Y
```

(Lines 10 and 50 set up a loop for Y. Lines 20 and 40 set up a loop for X. Line 30 SETS the graphic display.)

```
 60 FOR Y = 3 TO 7
 70  FOR X = 10 TO 30
 80   RESET (X,Y)
 90  NEXT X
100 NEXT y
```

(Lines 60 and 100 set up a loop for Y. Lines 70 and 90 set up a loop for X. Line 80 turns off the graphic display.)

RUN the program. The READY goofs up the display. How to get rid of it?

How about a timer loop? What's that? Just a loop with no printing; no function but to keep the Computer busy counting while we look at our display. Try this:

```
110 FOR A = 1 to 5000
120 NEXT A
```

(This tells the Computer to count from 1 to 5000 for A. We did not tell it to print A ... or to do anything. It is just going to be busy for a while.)

RUN this program now.

Random Number Generator

One of the most interesting functions of the TRS-80 is its ability to generate random numbers. Try this:

```
   10 X = RND (10)

   20 PRINT X

   30 GOTO 10
```

RUN it. Too much? BREAK the program and add:

```
    5 FOR N = 1 to 10
```

Change line 30:

```
   30 NEXT N
```

RUN it. Then add a comma at the end of line 20 and change the number for N so you can fill up the screen. Then, use a semi-colon and change N again to fill up the screen before the program ends.

Abbreviations

Most of the statements and commands can be abbreviated. This simplifies programming. Look at the section LISTING OF RADIO SHACK'S LEVEL I BASIC for all abbreviations, along with a complete listing of commands and statements and functions.

LISTING OF RADIO SHACK'S LEVEL I BASIC

Statement/ Abbreviation	Example
INPUT/IN.	INPUT X, Y, Z (Computer will request data for variables X, Y and Z) INPUT "ANY MESSAGE ON SCREEN"; X (Computer will print info inside quotes and request data for variable X)
DATA/D.	DATA X, Y, Z (holds data or values of variables for later READ statements)
READ (no abbr.)	READ X, Y, Z (instructs Computer to READ values assigned by DATA statements)
RESTORE/RES.	RESTORE (instructs Computer to start reading from first DATA statement)
PRINT/P.	PRINT X (Computer will print value of X) PRINT X, "ANY MESSAGE", Y, "ANY MESSAGE"; (, starts printing in next 16-character zone) (; prints with no spacing) (; at end of PRINT statement ties in the next line ie. suppresses carriage return)

Print Modifiers

PRINT AT/P. AT	PRINT AT 123, X, Y, Z PRINT AT X, "ANY MESSAGE" PRINT AT X + Y (begins printing at the location defined by the integer, variable or expression. It must be in the range of 0 to 1023)

PRINT TAB/P. TAB	PRINT TAB (12), PRINT TAB (X), PRINT TAB (X + Y), (begins printing at the number of spaces from the left as defined by the integer, variable or expression. Must be in the range of 0 to 63)
FOR/F.	FOR X = Y TO Z (STEP) Y (defines a series of values for X from variable Y to variable Z in steps equal to Y. If STEP is not specified, 1 is assumed)
NEXT/N.	NEXT X (used with FOR to complete a loop defining series for X)
STOP(no abbr.)	STOP (used to stop a program)
END (no abbr.)	END (optional, to end a program)
GOTO/G.	GOTO 1000 (instructs Computer to go to line 1000)
GOSUB/GOS.	GOSUB 3200 (instructs Computer to go to a subroutine program which starts at 3200; when it comes to a RETURN statement, the Computer will return to the main program)
RETURN/RET.	RETURN (used at end of subroutine program to return the Computer to the next line in the main program)

IF	IF X + Y = Z THEN 1000 (Note no punctuation. Conditional statement; if true, Computer proceeds to 1000; if false, Computer continues to next line)
	IF (X < Y) + (X <= Z) THEN 1000 (equations in parentheses are evaluated and return a 1 for true and 0 for false. Then operation between parenthesis is performed. Non zero proceeds to 1000, a zero goes to the next line number)
ON	ON X GOTO 1000, 2000, 3000 (multiple-way branch. Variable is evaluated and if X=1, Computer goes to 1000, if X=2 goes to 2000, etc.)
	ON X GOSUB 1000, 2000, 3000 (multiple-way branch to subroutine programs. If X=4, Computer has no where to go and stops)
CLS	(clears the screen)
LET	(Optional. Computes and assigns value)
SET/S.	SET (X, Y) (turns on graphics coordinate.Value of X to be 0-127; value of Y to be 0-47)
RESET/R.	RESET (X, Y) (turns off graphics coordinate. ie. "erases" a SET)
LIST/L.	(Not used in a program. Use to present a listing of program in the Computer)
RUN/R.	(Not used in a program. Causes program to run)
CONT(no abbr.)	(Not used in a program. Continues program after a BREAK key entry)

NEW/N.	(Not used in a program. Clears out any program from the Computer's memory. Always use before loading a new program)

FUNCTIONS

ABS	ABS(X) ABS(X+Y)	(returns or determines the absolute value of the variable or expression)
INT	INT(X) INT(X+Y)	(returns or determines the integer value of variable or expression. ie. "cuts off" any decimal fraction of that number)
RND	RND(100)	(returns or generates a random number between 1 and 100)
	RND(0)	(returns or generates a random number between 0 and 1)
	RND(X)	(returns or generates a random number with an upper limit equal to the value of X)
MEM/M.		(used to display the number of bytes left in the Computer's memory)
POINT	IF ... POINT(X,Y)	(used to check if graphics coordinate is on or off. If on,returns a 1; if off, returns a 0)

MATH OPERATORS

+	add (or logical "or")	=	equals
-	subtract or negation	>	greater than
*	multiply (or logical "and")	<	less than
/	divide	<=	less than or equal to
<>	not equal to	>=	greater than or equal to

VARIABLES

A through Z	Up to 6 significant digits
A(n)	4 bytes per subscript, from 0 to MEM/4
A$ and B$	Character strings up to 16 characters

SYNTAX

A colon (:) must separate all statements on the same line (except the first statement following an IF statement; in this case no colon is allowed).

All parentheses, commas and semi-colons used in the examples, must be used exactly as indicated. Improper punctuation will result in an error message or improper execution of the program.

All lines must begin with a line number. Information on a line may be up to 72 characters long. In this case, the screen display will overflow onto the next line, but the Computer will interpret it as one line of information.

LIMITS AND EXCEPTIONS

Line numbers must not exceed 32,767. A$ and B$ can not be used in expressions causing logical equations (eg. Computer can not compare A$ with B$).

ERROR MESSAGES

WHAT?	The Computer does not understand the instruction. This can be caused by misspelling or improper syntax or punctuation.
HOW?	The Computer understands the instruction but can not execute it. This can be caused by a GOTO with no line number to branch to, an integer wich exceeds 32,767 or a divide by 0.

SORRY   The Computer has run out of memory.

DATA FILES

Variables can be written to Cassette Tape with a PRINT # instruction. For example: to write A, B and C to tape

```
100 PRINT # A; ",";B; ",";C
```

This will write the values of A, B and C to tape. Note that each variable must be separated by ;",";. This is necessary in order to write commas to tape.

To read back the data, use the INPUT # instruction.

```
200 INPUT # A, B, C
```

This will read into the variables A,B and C the values written in the example above.

NOTE: To insure error-free Data Filing, always INPUT # the same number of variables as you PRINT #. If you print 3, input 3 (otherwise errors may occur or data may be lost).

MISCELLANEOUS

Remarks and documentation can be included in a program line if the line begins with the statement REM (remark). The information in a REM line will not be executed; it is a form of "note pad".

To erase an error, use the backspace key (←); this moves the cursor back one space for each pressing and erases information. To erase an entire line, merely enter a new line (retype the line with corrections, or enter a new line number with no information).

If a Listing is too long to be displayed on the screen, use the up-scroll arrow (↑). This key can not be used in any other instance.

The cursor (_) indicates the position on the screen where the next entry will be displayed. The cursor can be moved with the space bar.

When the Computer displays

```
READY

>_
```

it indicates it is ready for entries; "it's your turn". The prompt (>) indicates the Computer is ready for a line number (or other command).

If the Computer fails to turn the Cassette off at the end of a CLOAD or CSAVE, you must use the Reset button (inside the door at the left rear of the Keyboard Assembly). The Reset button can also be used to remove the Computer from an endless loop (or other "hang-up").

SOME JUST FOR FUN PROGRAMS

Try these various programs. They illustrate a number of the principles of the operation of your TRS-80. We hope it will stir your imagination and creativity ... so you go on from here to make up your own.

Graphics and RND function

```
10 FOR X = 0 TO 127 STEP N
20 N = RND(5)
30  FOR Y = 0 TO 47 STEP 5
40   SET (X, Y)
50  NEXT Y
60   FOR A = 0 TO 100
70   NEXT A
80 NEXT X
90 GOTO 10
```

```
10 X = RND(127)
20 Y = RND(47)
30 SET (X, Y)
40 GOTO 10
```

More Graphics

```
10 A = 0
20 B = 0
30 A = A + 1
40 B = B + 1
50 SET (A, B)
60 GOTO 30
```

Guess My Number

```
  5 LET Y = 0
  6 LET F = 0
 10 CLS
 20 P."                         GUESS MY NUMBER"
 30 P.:P."+++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++"
 40 P.:P."WHAT IS YOUR NAME? (FIRST NAME ONLY)":IN.A$
 50 CLS:P.:P.:P.:P."HELLO,";A$;"!"
 60 P.:P."    YOU ARE ABOUT TO PLAY A FUN NUMBER GAME."
 70    P."THE OBJECT OF THE GAME IS FOR YOU TO GUESS"
 80    P."THE NUMBER I AM THINKING OF.  IT IS FROM 0
 90    P."TO 100. (ENTER 0 WHEN READY)"
105 IN.P: GOTO 110
110 LET X = RND(100)
115 CLS:P.:P.:P.
117 LET F = 0: LET Y = 0
120 LET F=F+1:P.:P."ENTER YOUR ";F;"GUESS."
121 IN.A
123 LET Y = Y+1
125 P.
126 P.
130 IF A=X THEN 200
135 IF Y=10 THEN 350
140 IF A>X THEN 190
150 IF A<X THEN 180
180 P."HIGHER, ";A$;".":GOTO 120
190 P."TOO HIGH, ";A$;".": GOTO 120
```

```
200 CLS:P.:P.:P."YOU MADE IT IN";F;"TRIES!!!"

300 G. 370

350 CLS:P.:P.:P."       SORRY, ";A$;", BUT YOUR TEN TRIES"

360             P."ARE UP. THE NUMBER WAS";X;"."

370 P.:P."WOULD YOU LIKE TO PLAY AGAIN? (1=YES 0=NO)":IN.G

380 IF G=1 THEN 110

390 CLS:P.:P.:P.:P.:P."                                  RATS!!"

END
```

Printing a Large "X" with TAB Function

```
 5 CLS

10 X = 0

20 Y = 60

30 Y = Y-4    56 52 48 44 40 36 32 28 24 20

40 X = X+4     4  8 12 16 20 24 28 32 36 40

50 IF Y = 0 THEN 140

60 IF X   28 THEN 100

70 IF X >= 28 THEN 120

80 PRINT TAB(X);"****";TAB(Y);"****"

90 GOTO 30

100 PRINT TAB(X);"****";TAB(Y);"****"

110 GOTO 30

120 PRINT TAB(Y);"****";TAB(X);"****"

130 GOTO 30

140 FOR A = 0 TO 5000

150 NEXT A
```

Multiplication Tables (up to 12 x 12)

```
 10 PRINT
 20 A = RND (12)
 30 B = RND (12)
 40 PRINT
 50 PRINT A ; "*" ; B ; "= WHAT?"
 60 INPUT C
 70 IF C = A * B GOTO 120
 80 CLS
 90 PRINT "TRY AGAIN"
100 PRINT
110 GOTO 40
120 CLS
130 PRINT "CORRECT!!!"
140 GOTO 10
```

To Use

RUN and then enter the answer to the Multiplication Question on the screen.